REMARKS

Claims 1-36 remain pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Section 103(a) Rejections:

The Examiner rejected claims 1-5, 12-20, 23-31 and 34-36 under 35 U.S.C. § 103(a) as being unpatentable over Lee (U.S. Publication 2002/0059348), claims 6-8 as being unpatentable over Lee in view of in view of Lorenz (U.S. Patent 6,405,366), claims 9-11 as being unpatentable over Lee in view of Lorenz and further in view of Stern (U.S. Patent 6,507,855), and claims 21, 22, 32 and 33 as being unpatentable over Lee in view of Stern. Applicant respectfully traverses these rejections for at least the reasons stated below.

In regard to claim 1, contrary to the Examiner's assertion, the cited art does not teach or suggest a **software** documentation generator configured to input a **plurality of sources related to a software program**, where the plurality of sources include a plurality of different types of sources and include **at least one of a software library documentation file and software program source code**, as recited in claim 1. The Examiner refers to Lee at [0003], [0017], and [0020] as teaching this aspect of Applicant's claim. However, Lee fails to teach several of the elements contained in this portion of Applicant's claim, as shown in the following paragraphs.

First, Lee does not teach or suggest a **software** documentation generator. Instead, Lee's method generates documentation **for a product design**. The **source files** of Lee define a product design, and they **all originate from design tools** [Abstract, [0017]]. **Each source file** in Lee is associated with a product design [Abstract, 0020]. The method of Lee extracts comments from source files in order to generate **product design documentation** including the comments [[0003], [0006], claim 1], *not* to generate software documentation for a software program.

Second, Lee does not teach or suggest inputting a plurality of sources related to a software program. The Examiner admits that the plurality of sources in Lee are not related to a software program. Examiner seeks to remedy this deficiency by asserting that it would be obvious "to use the invention of Lee to produce the documentation for software products that can generate a documentation file from different types of source files." Examiner makes this assertion without offering any teaching, suggestion, motivation, or any other reason beyond his own opinion, stated with hindsight, that it would be obvious to use the invention of Lee in the manner proposed. No evidence of record has been provided to support this assertion. Furthermore, the Examiner suggests producing "documentation for software products that can generate a documentation file from different types of source files," which does not coincide with the limitations set forth in Applicant's claim. Accordingly, the Examiner has failed to establish a *prima facie* case of obviousness.

Third, Lee does not teach or suggest that the plurality of sources include at least one of a **software library documentation file** and **software program source code**. The Examiner cites paragraphs [0017] and [0020] as teaching this aspect of Applicant's claim. However, paragraph [0017] describes hardware design tools which create design source files that can be fed into the documentation tool to produce design documentation. Paragraph [0020] describes how the documentation tool extracts comments from the design source files, responsive to a configuration file containing parameters that define attributes of the product design associated with each source file. Neither paragraph makes any reference either to a software library documentation file or to software program source code. In connection with this aspect of the Applicant's claim, the Examiner writes "e.g. a software program that operates on the plurality of source files and parameters define the type of source file." It is not clear what the Examiner means by this. Lee does state in paragraph [0017] that the documentation tool itself is "a software program that operates on the plurality of source files regardless of the originating design tool." But this has no bearing on the limitation of Applicant's claim reciting that the plurality of sources include at least one of a **software library**

documentation file and **software program source code**. Applicant notes a very important distinction arising in connection with this aspect of claim 1. The term "software program source code" is well-known by those skilled in the art to mean source code written in a programming language, and subsequently used as source input into an already-executable program, such as a language compiler, to produce a new executable software program derived from the source code. It is well known that software program source code is written in order to create a new executable software program. For example, software program source code may be written in the C++ programming language for subsequent compilation by a C++ language compiling program into an executable software program. When Lee remarks that the documentation tool itself is "a software program that operates on the plurality of source files," Lee is not referring to "software program source code" in the well-known, canonical sense just described. The source files mentioned in Lee are never described as software program source code. To the contrary, the only example given in Lee is source files for a hardware circuit (ASIC). Neither, as the Examiner admits on page 4, paragraph 7, does Lee ever indicate a software library documentation file.

Further in regard to claim 1, contrary to the Examiner's assertion, the cited art does not teach or suggest the software documentation generator **analyzes the sources to identify a type for each one**, as recited in claim 1. The Examiner refers to [0063], lines 11-12, and [0020] as teaching this aspect of Applicant's claim. However, at lines 11-12 of paragraph [0063], Lee only indicates that "the tool makes a list of all source files of each specific type included in the configuration file." There is no suggestion that the tool **analyzes the sources to identify a type for each one**, only that it makes a list of all the source files of each type included in the configuration file. In fact, the cited paragraph [0020] explicitly states that the tool "operates responsive to parameters included in configuration file 204," and that those parameters include source file type. Paragraph [0021] teaches that the source file type parameter found in the configuration file defines source file extensions. For example, the tool would find and list VHDL files by searching for files with the extension "vhd." Thus, the tool of Lee uses type parameters found in the configuration file in order to locate and list files as indicated at lines 11-12

of paragraph [0063]. Lee's tool does not analyze the sources themselves to identify their types.

Further in regard to claim 1, contrary to the Examiner's assertion, the cited art does not teach or suggest the software documentation generator **extracts information** from the sources **based on the type of the source**, as recited in claim 1. The Examiner refers to [0063], lines 13-16 and [0021] as teaching this aspect of Applicant's claim. However, at lines 13-16, of paragraph [0063], Lee indicates only that the tool extracts comments from source files by parsing the source files for keywords. There is no suggestion that the tool extracts comments based on the type of the source. Paragraph [0021] only describes Fig. 3, which shows examples of parameters contained in the configuration file. As explained earlier, paragraph [0021] teaches that the source file type parameter found in the configuration file defines source file extensions, such as "vhd," which are used by the tool to locate and list files of that type. There is no indication, either here or elsewhere in Lee, that the tool **extracts information** from the sources **based on the type of the source**. In fact, the documentation tool of Lee extracts comments from the source files based on a user-created configuration file containing parameters [Abstract, 0020, 0026, 0029].

Further in regard to claim 1, contrary to the Examiner's assertion, the cited art does not teach or suggest the software documentation generator **aggregates the extracted information into a uniform format**, as recited in claim 1. The Examiner refers to [0033] as teaching this aspect of Applicant's claim. However, paragraph [0033] only indicates that the created output files might have a cover page, table of contents, sections, revision page, and the like, and that each section, in turn, might include a section title used to create a link from the table of contents to the corresponding section. There is no suggestion whatsoever that extracted information is aggregated into a uniform format. In fact, paragraph [0051] teaches that the user indicates which output files the documentation tool should generate, and that these might have a plurality of formats, such as HTML and PDF.

Further in regard to claim 1, contrary to the Examiner's assertion, the cited art does not teach or suggest the software documentation generator **transforms the aggregated information into specified sets of software documentation for the software program**, as recited in claim 1. The Examiner refers to [0051], lines 1-4 as teaching this aspect of Applicant's claim. However, the cited lines of paragraph [0051] teach that the user indicates which output files the documentation tool should generate, and that these might have a plurality of formats, such as HTML and PDF. There is no suggestion that the generated output files of Lee are sets of **software documentation for the software program** generated from information extracted from source files for the software program. Indeed, as indicated earlier, the **source files** of Lee define a product design, and they **all originate from design tools** [Abstract, [0017]]. **Each source file** in Lee is associated with a product design [Abstract, 0020]. The method of Lee extracts comments from source files in order to generate **product design documentation** including the comments [[0003], [0005], [0006], claim 1], *not* to generate software documentation for a software program.

In the Response to Arguments section of the Office Action, Examiner writes "Lee teaches in paragraph [0017] lines 13-15; the documentation tool is a software program that operates on the plurality of source files further in paragraph [0020] Lee teaches parameters includes types of the source file; Thus teaches plurality of sources comprises different types of sources and comprises one or more of program source code." However, as already shown in detail above, the source files in Lee are not software program source code. Examiner further writes "Although Lee teaches one example wherein the source files are hardware description files, Lee does not limit the source files to such." This is factually incorrect, as already discussed. Throughout the entire disclosure, Lee makes it clear that the source files are produced by design tools. For example, Lee teaches that the source files define a design and originate from design tools [Abstract]. At [0017], writes "Design tools 110, e.g., tools 110i . . . 110n, create a corresponding plurality of source files 114, e.g., source files 114i-114n, defining a product design. [Fig. 1A]." At [0020], Lee discloses that the documentation tool operates responsive to parameters included in the configuration file where the parameters define

attributes of the product design associated with each source file." The entire keyword syntax disclosed by Lee on page 3 is directed at product design. For example, keyword REGISTER is used when registers are included in the product design. Moreover, **it is the Examiner who has the burden** to show that **every** claim limitation is taught or suggested by the prior art. *In re Warner*, 154 USPQ 173, 177 (C.C.P.A. 1967), *cert. denied*, 389 U.S. 1057 (1968). Even if Lee was not explicitly limited to hardware product designs, that does not mean that Lee teaches or suggests Applicant's claimed invention either. The Examiner's assertion in the regard is completely unsupported by any actual evidence of record. Therefore, the Examiner has clearly failed to establish a *prima facie* rejection.

Independent claims 15 and 26 include limitations similar to independent claim 1, and so the arguments presented above apply with equal force to these claims, as well. For at least the reasons given above, the cited art cannot be said to anticipate Applicant's claims 1, 15 and 26.

In regard to claim 2, contrary to the Examiner's assertion, the cited art does not teach or suggest the system as recited in claim 1, where one of the sources includes the software library documentation file, as recited in claim 2. The Examiner refers to Lee at [0003], lines 5-7 and [0017], lines 13-15 as teaching this aspect of Applicant's claim, asserting that Lee "inherently teaches" this because the documentation tool operates on a plurality of source files. However, Lee's paragraph [0003] merely places Lee's invention within the general sphere documentation generation, disclosing that the invention *relates* "to a method of automatically generating documentation from a variety of source files." As has been thoroughly described earlier, and as the cited lines in paragraph [0017] reveal, the plurality of source files are all produced by a design tool. There is no indication anywhere in Lee of a software library documentation file. For at least these reasons, the cited art cannot be said to anticipate Applicant's claim 2.

In regard to claim 3, contrary to the Examiner's assertion, the cited art does not teach or suggest the system as recited in claim 2, where the software library

documentation file includes a tag library descriptor (TLD), as recited in claim 3. The Examiner refers to Lee at [0003], lines 5-7 and [0017], lines 13-15 as teaching this aspect of Applicant's claim, asserting that Lee "inherently teaches" this because the documentation tool operates on a plurality of source files. However, Lee's paragraph [0003] merely places Lee's invention within the general sphere documentation generation, disclosing that the invention *relates* "to a method of automatically generating documentation from a variety of source files." As has been thoroughly described earlier, and as the cited lines in paragraph [0017] reveal, the plurality of source files are all produced by a design tool. There is no indication anywhere in Lee of a software library documentation file which includes a tag library descriptor (TLD). For at least these reasons, the cited art cannot be said to anticipate Applicant's claim 3.

In regard to claim 5, contrary to the Examiner's assertion, the cited art does not teach or suggest the system as recited in claim 1, where the documentation sets comprise documentation for one or more application programming interfaces (API) provided by a software library, as recited in claim 5. The Examiner refers to Lee at [0003], lines 5-7 and [0017], lines 13-15 as teaching this aspect of Applicant's claim, asserting that Lee "inherently teaches" this because the documentation tool operates on a plurality of source files. However, Lee's paragraph [0003] merely places Lee's invention within the general sphere documentation generation, disclosing that the invention *relates* "to a method of automatically generating documentation from a variety of source files." As has been thoroughly described earlier, and as the cited lines in paragraph [0017] reveal, the plurality of source files are all produced by a design tool. There is no indication anywhere in Lee that the specified *output* files of the documentation tool include documentation for application programming interfaces (API) provided by a software library. For at least these reasons, the cited art cannot be said to anticipate Applicant's claim 5.

In regard to claim 6, contrary to the Examiner's assertion, the cited art does not teach or suggest the system as recited in claim 1, where the software documentation generator is configured to include input source plug-ins, where each input source plug-in

corresponds to one or more of the source file types and each input source plug-in is configured to extract information from source files of types to which the plug-in corresponds, as recited in claim 6. The Examiner admits that Lee does not teach this, but asserts that Lee teaches at [0020-0021] that "the software documentation generator is configured to include each input source is configured to extract information from source files to which it corresponds." It is not clear what the Examiner means by this. Examiner further asserts that "Lorenz teaches one or more source plug-ins, wherein each input source plug-in corresponds to one or more of the source file types." However, the plug-ins of Lorenz are not inputs to a software documentation generator which correspond to input source file types. Instead, the plug-ins of Lorenz are modular *interface* plug-ins which *access data* in various data formats, *not* input source plug-ins that correspond to certain source file types.

Further in regard to claim 6, the Examiner asserts that it would be obvious to include Lorenz's plug-ins in Lee's system "so that Lee's system can operate on [a] variety of source files originating from different types of sources." However, Lee already accomplishes this goal without the use of plug-ins, as disclosed in portions of Lee already cited by the Examiner [0003, 0063]. Thus there would be no need to augment Lee with the plug-ins of Lorenz. Furthermore, the plug-ins of Lorenz are not inputs to a software document generator, but rather modular *interface* plug-ins which *access data* in various data formats. Examiner has failed to explain *how* to incorporate the plug-ins of Lorenz into the invention of Lee. Moreover, the inventions of Lee and Lorenz belong to disparate fields. Thus, one of ordinary skill would not have combined the teachings of Lee with the teachings of Lorenz as proposed by the Examiner. Accordingly, the Examiner has failed to establish a *prima facie* case of obviousness. For at least these reasons, the cited art cannot be said to anticipate Applicant's claim 6.

Applicants also assert that numerous others of the dependent claims recite further distinctions over the cited art. However, since the rejections have been shown to be unsupported for the independent claims, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5681-75900/RCK.

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
Attorney for Applicant

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: June 30, 2008